Intro. To Deep Learning

Phuriwat Angkoondittaphong

What is deep learning

- Deep learning is subset of machine learning that use Artificial Neural Network (ANN).
- Machine learning is an AI algorithm that can learn from data using statistic, or something else.

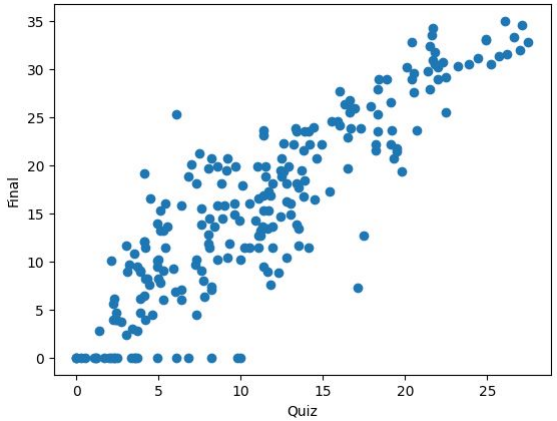
How (mathematical) model learn

- Data
- Example model
- Objective function
- Optimization
  - Analytically
  - Numerically

What is Data

- Consider regression problem

$$D = \{(x_1, y_1), (x_2, y_2), ..., \}$$
$$\text{where } x_i, y_i \in \mathbb{R}$$



Consider a linear regression model

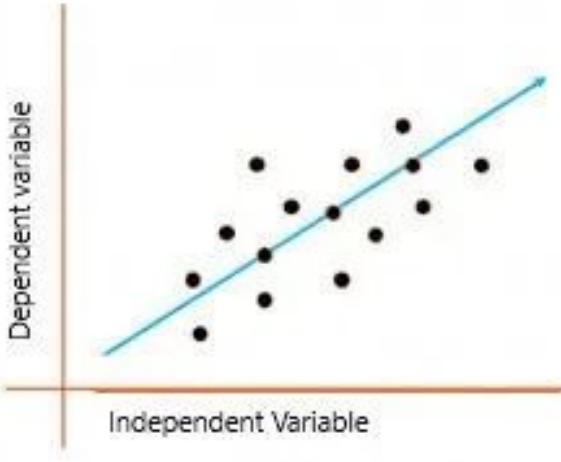
$$M : \mathbb{R} \to \mathbb{R}$$
$$M(x) = wx + b$$

- The model has 2 parameters, w and b
- By adjusting, it means to change parameters so that it fit to the training data more
- M is estimator, meaning, the model tried to estimate some value on the real line.

How to indicate “best” model (Objective function)

- how do we know that the model perform good or bad
- One idea is to sum up the error
- One of them called sum of squared error

$$e = \sum_i (y_i - M(x_i))^2$$

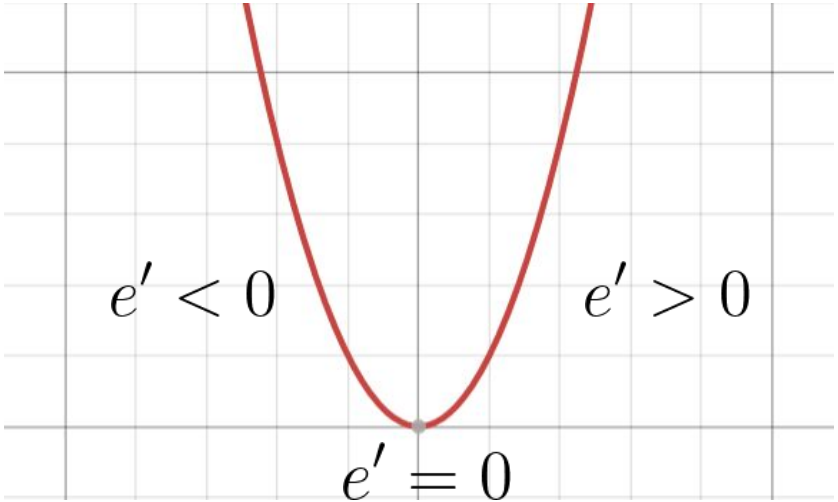


Then we can find m and b such that produce minimum e

$$(\hat{m}, \hat{b}) = \text{argmin}_{(m,b)} e$$

Optimizing Linear Regression Analytically

- Since, the error function is quite simple, we can solve for a minima in the function using calculus of variations. Leave this as exercise for viewer.



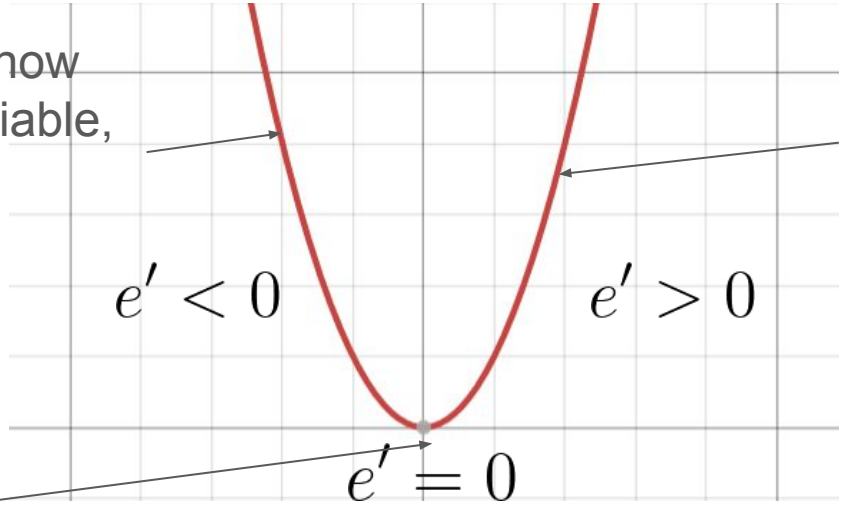
At e’ = 0, the e will be its minimum

Optimizing Linear regression: Deep learning perspective

- But in deep learning, we usually use model that is more complex than this
- Something like 1M of parameters, which can be hard to calculate derivative analytically.
- However, we can calculate it numerically.
- The algorithm is called Gradient descent

Graph of e over a variable

If e’ at this point, we know that if we increase variable, e will be reduce



If e’ at this point, we know that reduce variable to reduce e

Optimal point

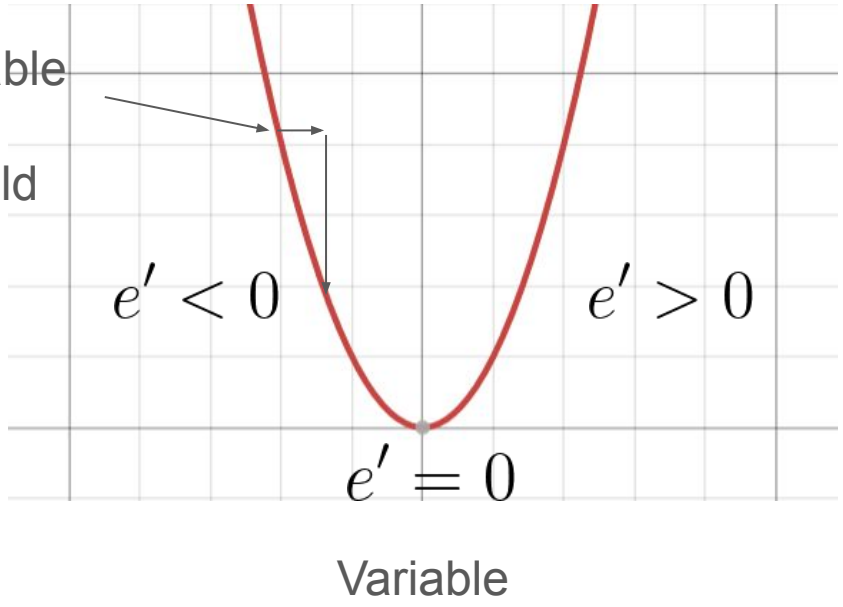
Variable

With this, the variable knows its direction that can make the model better

Learning rate

If e’ at this point, increase the variable to reduce e.
We don’t know how much we should take a step.

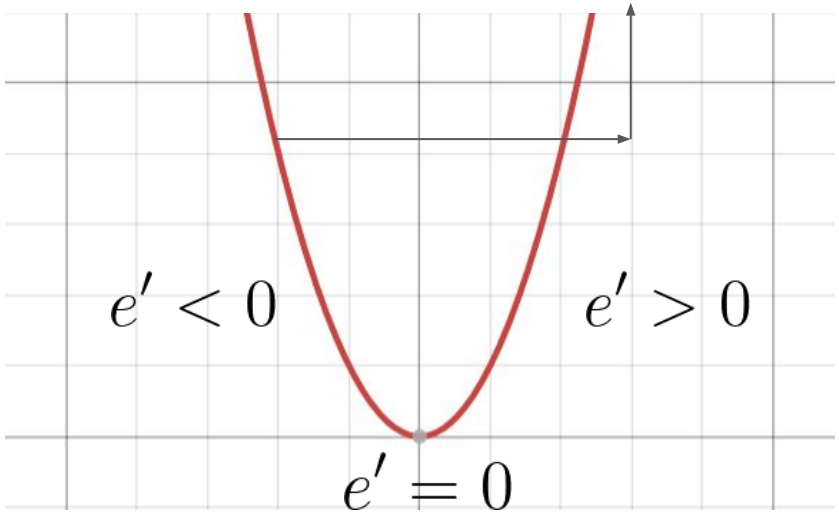
Example of a step



Learning rate

The step should be
not too large, not too small

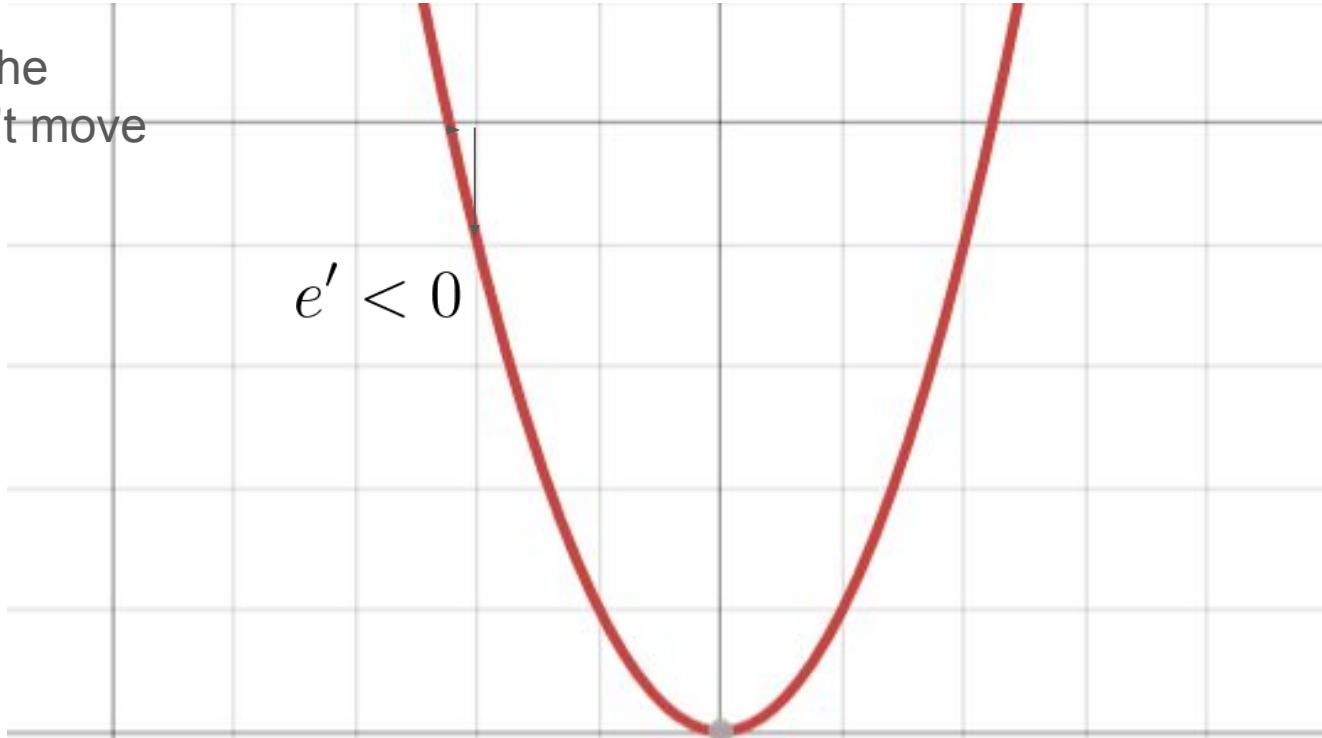
Example of large step



Learning rate

Too small and the parameter won't move

Example of small step



Learning rate

- Size of the step is usually called Learning rate (LR)
- It is one of the most important hyperparameter

How we calculate derivative (gradient)

- It use computational graph something, something, something, …

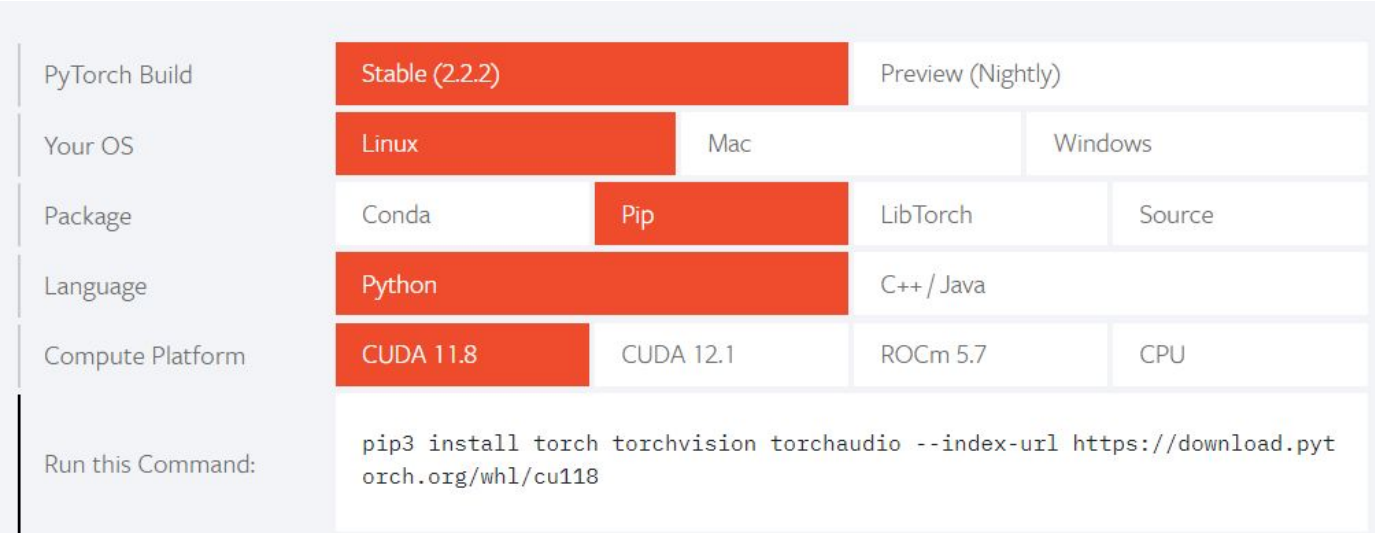
- A story for another day.
- Today we gonna use PyTorch to calculate for us

PyTorch

- PyTorch is a library for scientific computing.
- Written in C++ and CUDA with Python interface
- It provides fast multidimensional array computing functions (eg. matrix multiplication) on both CPU and GPU.
- It also provides easy way to compute gradient (automatic differentiation system).

PyTorch installation

- Installation (not recommended)



- If you dont want to install it yourself, just use google colab

PyTorch Basic Workshop

Train simple linear regression

	A	B
1	Quiz	Final
2	25.2	30.5
3	8.4	13.6
4	4.9	14
5	11.1	16.6
6	9	15.8
7	7.8	6.3
8	3	2.4
9	16	27.7
10	2.5	0
11	13.8	16.8
12	9.6	14.9
13	11.8	7.6
14	16.9	25.9

Model training in PyTorch

1. Create dataset
2. Define the model
3. Choose optimizer
4. Choose loss function
5. Train
6. Evaluate

Create dataset

```python
class ScoreDataset(torch.utils.data.Dataset):
    def __init__(self, x, y):
        self.x = x
        self.y = y
        assert len(x) == len(y)
        self.n = len(x)
    def __getitem__(self, idx):
        return self.x[idx], self.y[idx]
    def __len__(self):
        return self.n
```

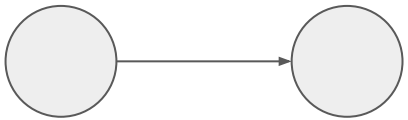
Define model

- In order to compute gradient, the model must consist of differentiable components.

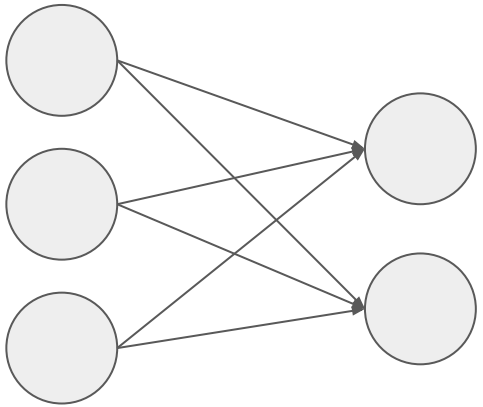
Model Layers: Linear Transformation

- Linear
  - The simplest layer of all nn.
  - “Linear” from Linear transformation
  - Simply Linear regression on steroid

$$y = Wx + B$$



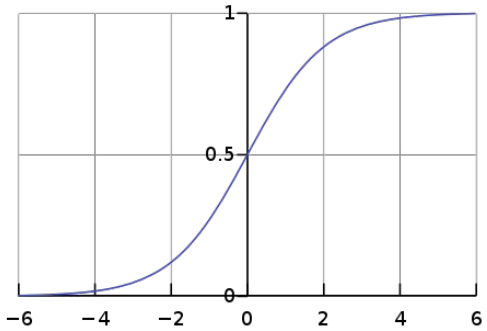
Linear regression with one input and one output

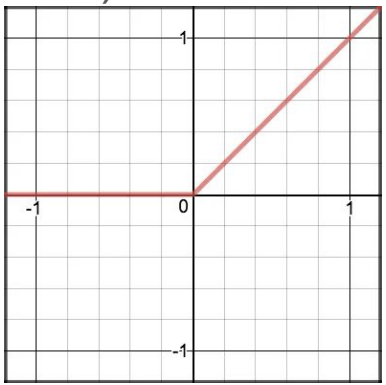


Linear regression with 3 inputs and 2 outputs

Model Layers: Activation function

- Chaining many linear layers do not work.
- We need non-linearity between linear layers.
- The most common one is ReLU = max(0, x)
- Sigmoid $\sigma(x) = \frac{1}{1 + e^{-x}}$
- SoftMax (usually used on the last layer of classification)





Training

```python
def train_model_one_epoch(model, train_dataloader, optimizer, loss_fn):
    cummulative_loss = 0
    for data in train_dataloader:
        # Every data instance is an input + label pair
        inputs, labels = data

        # Zero your gradients for every batch!
        optimizer.zero_grad()

        # Make predictions for this batch
        outputs = model(inputs)

        # Compute the loss and its gradients
        loss = loss_fn(outputs, labels)
        loss.backward()

        # Adjust learning weights
        optimizer.step()

        cummulative_loss += loss.item()

    cummulative_loss /= len(train_dataloader)
    return cummulative_loss
```

Train simple classifier